Internet Engineering Task Force G. Montenegro
INTERNET DRAFT Sun Microsystems, Inc.
S. Dawkins
Nortel Networks
M. Kojo
University of Helsinki
V. Magret
Alcatel
November 18, 1998

Long Thin Networks
draft-montenegro-pilc-ltn-00.txt

Status of This Memo

Abstract

In view of the unpredictable and problematic nature of long thin networks (for example, wireless WANs), arriving at an optimized transport is a daunting task. We have reviewed the existing proposals along with future research items. Based on this overview, we also recommend mechanisms for implementation in long thin networks.

Our goal is to identify a TCP that works for all users,including users of long thin networks. We started from the working recommendations of the IETF TCP Over Satellite Links (tcpsat) working group with this end in mind.

We recognize that not every tcpsat recommendation will be required for long thin networks as well, and are working toward a set of TCP recommendations that are "benign" in environments that do not require them.

Table of Contents

1 Introduction

Optimized wireless networking is one of the major hurdles that Mobile Computing must solve if it is to enable ubiquitous access to networking resources. However, current data networking protocols have been optimized primarily for wired networks. Wireless environments have very different characteristics in terms of latency, jitter, and error rate as compared to wired networks. Accordingly, traditional protocols are ill-suited to this medium.

Mobile Wireless networks can be grouped in W-LANs (for example, 802.11 compliant networks) and W-WANs (for example, CDPD [CDPD], Ricochet, CDMA [CDMA], PHS, DoCoMo, GSM [GSM] to name a few). W-WANs present the most serious challenge, given that the length of the wireless link (expressed as the delay*bandwidth product) is typically 4 to 5 times as long as that of its W-LAN counterparts. For example, for an 802.11 network, assuming the delay (round-trip time) is about 3 ms. and the bandwidth is 1.5 Mbps, the delay*bandwidth product is 4500 bits. For a W-WAN such as Ricochet, a typical round-trip time may be around 500 ms. (the best is about 230 ms.), and the sustained bandwidth is about 24 Kbps. This yields a delay*bandwidth product roughly equal to 1.5 KB. In the near future, 3rd Generation wireless services will offer 384Kbps and more. Assuming a 200 ms round-trip, the delay*bandwidth product in this case is 76.8 Kbits (9.6 KB). This value is larger than the default 8KB buffer space used by many TCP implementations. This means that, whereas for W-LANs the default buffer space is enough, future W-WANs will operate inefficiently (that is, they will not be able to fill the pipe) unless they override the default value. A 3rd Generation wireless service offering 2 Mbps with 200-millisecond latency requires a 50 KB buffer.

Most importantly, latency across a link adversely affects throughput. For example, [MSMO97] derives an upper bound on TCP throughput. Indeed, the resultant expression is inversely related to the round-trip time.

The long latencies also push the limits (and commonly transgress them) for what is acceptable to users of interactive applications.

As a quick glance to our list of references will reveal, there is a wealth of proposals that attempt to solve the wireless networking problem. In this document, we survey the different solutions available or under investigation, and issue the corresponding recommendations.

There is a large body of work on the subject of improving TCP performance over satellite links. The documents under development by the tcpsat working group of the IETF [AGS98, ADGGHOSSTT98] are very relevant. In both cases, it is essential to start by improving the characteristics of the medium by using forward error correction (FEC) at the link layer to reduce the BER (bit error rate) from values as high as 10-3 to 10-6 or better. This makes the BER manageable. Once in this realm, retransmission schemes like ARQ (automatic repeat request) may be used to bring it down to zero. Notice that sometimes it may be desireable to forgo ARQ because of the additional delay it implies. In particular, time sensitive traffic (video, audio) must be delivered within a certain time limit beyond which the data is obsolete. Exhaustive retransmissions in this case merely succeed in wasting time in order to deliver data that will be discarded once it arrives at its destination. This indicates the desireability of augmenting the protocol stack implementation on devices such that the upper protocol layers can inform the link and MAC layer when to avoid such costly retransmission schemes.

Networks that include satellite links are examples of "long fat networks" (LFNs or "elephants"). They are "long" networks because their round-trip time is quite high (for example, 0.5 sec and higher for geosynchronous satellites). Not all satellite links fall within the LFN regime. In particular, round-trip times in a low-earth orbiting (LEO) satellite network may be as little as a few milliseconds (and never extend beyond 160 to 200 ms). W-WANs share the "L" with LFNs. However, satellite networks are also "fat" in the sense that they may have high bandwidth. Satellite networks may often have a delay*bandwidth product above 64 KBytes, in which case they pose additional problems to TCP [TCPHP]. W-WANs do not generally exhibit this behavior. Accordingly, this document only deals with links that are "long thin pipes", and the networks that contain them: "long thin networks". We call these "LTNs".

This document does not give an overview of the API used to access the underlying transport. We believe this is an orthogonal issue, even though some of the proposals below have been put forth assuming a given interface. It is possible, for example, to support the traditional socket semantics without fully relying on TCP/IP transport [MOWGLI].

Our focus is on the on-the-wire protocols. We try to include the most relevant ones and briefly (given that we provide the references needed for further study) mention their most salient points.

1.1 Architecture

One significant difference between LFNs and LTNs is that we assume the W-WAN link is the last hop to the end user. This allows us to assume that a single base station sees all packets transferred between the wireless mobile device and the rest of the Internet. This is only one of the topologies considered by the TCP Satellite community.

Given our focus on mobile wireless applications, we only consider a very specific architecture that includes:

- a wireless mobile device, connected via

- a wireless link (which may, in fact comprise several hops at the link layer), to

- a base station (sometimes referred to as an intermediate agent) connected via

- a wireline link, which in turn interfaces with

- the landline Internet and millions of legacy servers and web sites.

Specifically, we are not as concerned with paths that include two wireless segments separated by a wired one. This may occur, for example, if one mobile device connects across its immediate wireless segment via a base station to the internet, and then via a second wireless segment to another mobile device. Quite often, mobile devices connect to a legacy server on the wired internet.

Typically, the endpoints of the wireless segment are the base station and the mobile device. However, the latter may be a wireless router to a mobile network. This is also important and has applications in, for example, disaster recovery.

Our target architecture has implications which concern the deployability of candidate solutions. In particular, an important requirement is that we cannot alter the networking stack on the legacy servers. It would be preferable to only change the networking stack at the base station, although changing it at the mobile devices is certainly an option and perhaps a necessity.

We envision mobile devices that can use the wireless medium very efficiently, but overcome some of its traditional constraints. That is, full mobility implies that the devices have the flexibility and agility to use whichever happens to be the best network connection available at any given point in time or space. Accordingly, devices could switch from a wired office LAN and hand over their ongoing connections to continue on, say, a wireless WAN. This type of agility also requires Mobile IP [RFC2002].

NOTE: Must we replace "base station" with some other term (e.g., LTN-edge device)? "Base station" is a good term when discussing W-LANs but a misleading one in almost all W-WAN environments. W-LANs, the wireless link is between mobile device and base station, but within a typical W-WAN infrastructure there are several wireline hops between the actual W-WAN base station and the W-WAN edge device that provides the connection point to the landline Internet. These "base stations" do not have an IP stack, so, for example, they cannot have a SNOOP module.

1.2 Assumptions about the Radio Link

The system architecture described above assumes at most one wireless link (perhaps comprising more than one wireless hop). However, this is not enough to characterize a wireless link. Additional considerations are:

- What are the error characteristics of the wireless medium? The link may present a higher BER than a wireline network due to burst errors and disconnections. The techniques below usually do not address all the types of errors. Accordingly, a complete solution should combine the best of all the proposals. Nevertheless, in this document we are more concerned with (and give preference to solving) the most typical case: (1) higher BER due to random errors (which implies longer and more variable delays due to link-layer error corrections and retransmissions) rather than (2) an interruption in service due to a handoff or a disconnection. The latter are also important and we do include relevant proposals in this survey.

- Is the wireless service datagram oriented, or is it a virtual circuit? Currently, switched virtual circuits are more common, but packet networks are starting to appear, for example, Metricom's Starmode [CB96], CDPD [CDPD] and General Packet Radio Service (GPRS) [GPRS],[BW97] in GSM.

- What kind of reliability does the link provide? Wireless services typically retransmit a packet until it has been acknowledged by the target. They may allow the user to turn off this behavior. For example, GSM allows RLP (Reliable Link Protocol) to be turned off. Metricom has a similar "lightweight" mode.

- Does the mobile device transmit and receive at the same time? Doing so increases the cost of the electronics on the mobile device. Typically, this is not the case. We assume the typical case in this document.

- Does the mobile device directly address more than one peer on the wireless link? Packets to each different peer may traverse spatially distinct wireless paths. Accordingly, the path to each peer may exhibit very different characteristics. Quite commonly, the mobile device addresses only one peer (the base station) at any given point in time. When this is not the case, techniques such as Channel-State Dependent Packet Scheduling come into play (see the section "Packet Scheduling" below).

2 Should it be IP or Not?

The first decision is whether to use IP as the underlying network protocol or not. In particular, some data protocols evolved from wireless telephony are not always -- though at times they may be -- layered on top of IP [MOWGLI, WAP]. These proposals are based on the concept of proxies that provide adaptation services between the wireless and wireline segments.

This is a reasonable model for mobile devices that always communicate through the proxy. However, we expect many wireless mobile devices to utilize wireline networks whenever they are available. This model closely follows current laptop usage patterns: devices typically utilize LANs, and only resort to dial-up access when "out of the office."

For these devices, an architecture that assumes IP is the best approach, because it will be required for communications that do not traverse the base station (for example, upon reconnection to a W-LAN or a 10BaseT network at the office).

2.1 Underlying Network Error Characteristics

Using IP as the underlying network protocol requires a certain (low) level of link robustness that is expected of wireless links.

IP, and the protocols that are carried in IP packets, are protected end-to-end by checksums that are relatively weak (and, in some cases, optional). For much of the internet, these checksums are sufficient; in wireless environments, the error characteristics of the raw wireless link are much less robust than the rest of the end-to-end path. This makes end-to-end detection of network errors undesirable, because damaged IP packets are propagated through the network only to be discarded at the destination host, and suggests that link-level mechanisms should be used to detect and correct transmission errors over these links.

A better approach is to use link-layer mechanisms such as FEC, retransmissions, and so on in order to improve the characteristics of the wireless link and present a much more reliable service to IP. This approach has been taken by CDPD, Ricochet and CDMA.

This approach is roughly analogous to the successful deployment of Point-to-Point Protocol (PPP), with robust framing and 16-bit checksumming, on wireline networks as a replacement for the Serial Line Interface Protocol (SLIP), with only a single framing byte and no checksumming.

[AGS98] recommends the use of FEC in satellite environments.

Notice that the link-layer could adapt its frame size to the prevalent BER. It would perform its own fragmentation and reassembly so that IP could still enjoy a large enough MTU size [LS98].

A common concern for using IP as a transport is the header overhead it implies. Typically, the underlying link-layer appears as PPP [RFC1661] to the IP layer above. This allows for header compression schemes [IPHC, IPHC-PPP] which greatly alleviate the problem.

2.2 Non-IP Alternatives

A number of non-IP alternatives aimed at wireless environments have been proposed. One representative proposal is discussed here.

2.2.1 WAP

The Wireless Application Protocol (WAP) specifies an application framework and network protocols for wireless devices such as mobile telephones, pagers, and PDAs [WAP]. The architecture requires a proxy between the mobile device and the server. The WAP protocol stack is layered over a datagram transport service. Such a service is provided by most wireless networks; for example, IS-136, GSM SMS/USSD, and UDP in IP networks like CDPD and GSM GPRS. The core of the WAP protocols is a binary HTTP/1.1 protocol with additional features such as header caching between requests and a shared state between client and server.

2.2.2 Deploying Non-IP Alternatives

IP is such a fundamental element of the internet that non-IP alternatives face substantial obstacles to deployment, because they do not exploit the IP infrastructure. Any non-IP alternative that is used to provide gatewayed access to the internet must map between IP addresses and non-IP addresses, must terminate IP-level security at a gateway, and cannot use IP-oriented discovery protocols (Dynamic Host Configuration Protocol, Domain Name Services, Lightweight Directory Access Protocol, Service Location Protocol, etc.) without translation at a gateway.

A further complexity occurs when a device supports both wireless and wireline operation. If the device uses IP for wireless operation, uninterrupted operation when the device is connected to a wireline network is possible (using Mobile IP). If a non-IP alternative is used, this switchover is more difficult to accomplish.

Non-IP alternatives face the burden of proof that IP is so ill-suited to a wireless environment that it is not a viable technology.

2.3 IP-based Alternatives

Given its worldwide deployment, IP is an obvious choice for the underlying network technology. Optimizations implemented at this level benefit traditional internet application protocols as well as new ones layered on top of IP or UDP.

2.3.1 Path MTU Discovery

Path MTU discovery benefits any protocol built on top of IP. It allows a sender to determine what the maximum end-to-end transmission unit is to a given destination. Withouth Path MTU discovery, the default MTU size is 512. The benefits of using a larger MTU are:

- Smaller ratio of header overhead to data

- Allows TCP to grow its congestion window faster, since it increases in units of segments.

Of course, for a given BER, a larger MTU has a correspondingly larger probability of error within any given segment. The BER may be reduced using lower level techniques like FEC and link-layer retransmissions. The issue is that now delays may become a problem due to the additional retransmissions, and the fact that packet transmission time increases with a larger MTU.

[AGS98] recommends use of Path MTU Discovery in satellite environments.

2.3.2 Non-TCP Proposals

Other proposals assume an underlying IP datagram service, and implement an optimized transport either directly on top of IP [NETBLT] or on top of UDP [MNCP]. Not relying on TCP is a bold move, given the wealth of experience and research related to it. It could be argued that the internet has not collapsed because its main protocol, TCP, is very careful in how it uses the network, and generally treats it as a black box assuming all packet losses are due to congestion and prudently backing off. This avoids further congestion.

However, in the wireless medium, packet losses may also be due to corruption due to high BER, fading, and so on. Here, the right approach is to try harder, instead of backing off. Alternative transport protocols are:

- NETBLT [NETBLT, RFC1986, RFC1030]

- MNCP [MNCP]

- ESRO [RFC2188]

- RDP [RFC908, RFC1151]

- VMTP [VMTP]

3 The Case for TCP

This is one of the most hotly debated issues in the wireless arena. Here are some arguments against it:

- It is generally recognized that TCP does not perform well in the presence of significant levels of non-congestion loss. TCP detractors argue that the wireless medium is one such case, and that it is hard enough to fix TCP. They argue that it is easier to start from scratch.

- TCP has too much header overhead.

- By the time the mechanisms are in place to fix it, TCP is very heavy, and ill-suited for use by lightweight, portable devices.

and here are some in support of TCP:

- It is preferrable to continue using the same protocol that the rest of the Internet uses for compatibility reasons. Any extensions specific to the wireless link may be negotiated.

- Legacy mechanisms may be reused (for example congestion control)

- Link-layer FEC and ARQ can reduce the BER such that any losses TCP does see are, in fact, caused by congestion (or a sustained interruption of link connectivity). Modern W-WAN technologies do this (CDPD, US-TDMA, CDMA, GSM), thus improving TCP throughput.

- Handoffs among different technologies are made possible by Mobile IP [RFC2002], but only if the same protocols, namely TCP/IP, are used throughout.

- Given TCP's wealth of research and experience, alternative protocols are relatively immature, and the full implications of their widespread deployment not clearly understood.

Overall, we feel that TCP is fixable. Mechanisms to do so are included in the next sections.

4 Candidate Optimizations

There is a large volume of work on the subject of optimizing TCP for operation over wireless media. Even though satellite networks generally fall in the LFN regime, our current LTN focus has much to benefit from it. For example, the work of the TCP-over-Satellite working group of the IETF has been extremely helpful in preparing this section [AGS98, ADGGHOSSTT98].

4.1 TCP: Current Mechanisms

A TCP sender adapts its use of bandwidth based on feedback from the receiver. The high latency characteristic of LTNs implies that TCP's adaptation is correspondingly slower than on networks with shorter delays. Delayed ACKs and small MTUs may slow adaptation even further.

4.1.1 Slow Start and Congestion Avoidance

Slow Start and Congestion Avoidance [RFC2001, updated in TCPCONG] are the basis for TCP's successful deployment throughout the internet. However there are two reasons why the wireless medium adversely affects them:

- Slow start is invoked whenever a loss is detected, assuming the network is congested. This is why it is important to minimize the losses caused by corruption, leaving only those that TCP expects.

- The sender increases its window based on the number of ACKs received. This rate, of course, is dependent on the RTT (round-trip-time) between sender and receiver, which implies long delays in high latency links like LTNs.

- During slow start, the sender increases its window in units of segments. This is why it is important to use an appropriately large MTU which, in turn, requires reliable link layers.

4.1.2 Fast Retransmit and Fast Recovery

Fast retransmit [RFC2001, updated in TCPCONG] allows the receiver to inform the sender (by sending several duplicate ACKs) of a lost segment. The sender retransmits what it considers to be this lost segment without waiting for the full timeout. This saves time.

After a fast retransmit, a sender invokes the fast recovery [RFC2001] algorithm, whereby it invokes congestion avoidance, but not slow start. This also saves time.

With LTN links, efficient recovery from multiple losses within a single window is more difficult to achieve, and waiting for three duplicate ACKs to arrive postpones retransmission noticeably.

Recommendation: Implement at this time. This is a widely-implemented optimization and is currently at Proposed Standard level. [AGS98] recommends implementation of Fast Retransmit/Fast Recovery in satellite environments.

4.2 Connection Setup with T/TCP [RFC1397, RFC1644]

TCP engages in a "three-way handshake" whenever a new connection is set up. Data transfer is only possible after this phase has completed successfuly. T/TCP allows data to be exchanged in parallel with the connection set up, saving valuable time for short transactions on long-latency networks.

Recommendation: T/TCP is not recommended, for these reasons:

- It is an Experimental RFC, and has not been advanced.

- It is not widely deployed, and it has to be deployed at both ends of a connection.

- Security concerns have been raised that T/TCP is more vulnerable to address-spoofing attacks than TCP itself.

- At least some of the benefits of T/TCP (eliminating three-way handshake on subsequent query-response transactions, for instance) are also available with persistent connections on HTTP/1.1, which is more widely deployed.

[ADGGHOSSTT98] does not have a recommendation on T/TCP in satellite environments.

4.3 Slow Start Proposals

Because slow start dominates the network response seen by interactive users at the beginning of a TCP connection, a number of proposals have been made to modify or or eliminate slow start in long latency environments.

Stability of the internet is paramount, so these proposals must demonstrate that they will not adversely affect internet congestion levels in significant ways. The needs of the many outweigh the needs of the few.

4.3.1 Larger Initial Window

Full slow start, with an initial window of one segment, is a time-consuming bandwidth adaptation procedure over LTNs. Recent proposals suggest starting off with an initial window larger than one segment [RFC2414, AHO98].

In simulations with an increased initial window of three packets [RFC2415], this proposal does not contribute significantly to packet drop rates, and it has the added benefit of improving initial response times when the peer device delays acknowledgements during slow start (see next proposal).

[RFC2416] addresses situations where the initial window exceeds the number of buffers available to TCP and indicates that this situation is no different from the case where the congestion window grows beyond the number of buffers available.

We expect the IETF tcp-impl working group to recommend allowing an initial window of at least two segments, and perhaps as many as four, in the near future, in environments where this significantly improves performance (LFNs and LTNs).

Recommendation: Implement this on devices now. The research on this optimization indicates that 3 segments is a safe initial setting, and is centering on choosing between 2, 3, and 4. For now, use 3 [RFC2415], which at least allows clients running query-response applications to get an initial ACK from unmodified servers without waiting for a delayed ACK timeout of 200-500 milliseconds, and saves two round-trips.

Much of the benefit of this optimization is also available (after the first request-response exchange) when clients and servers both implement HTTP/1.1. This optimization works with older servers that implement only HTTP/1.0.

4.3.2 Handling Acknowledgments During Slow Start

The sender increases its window based on the flow of ACKs coming back from the receiver. Particularly during slow start, this flow is very important. A couple of the proposals that have been studied are (1) ACK counting and (2) ACK-every-segment.

4.3.2.1 ACK Counting

The main idea behing ACK counting is:

- Make each ACK count to its fullest by growing the window based on the data being acknowledged (byte counting) instead of the number of ACKs (ACK counting). This has been shown to cause bursts which lead to congestion. [Allman98] shows that Limited Byte Counting (LBC), in which the window growth is limited to 2 segments, does not lead to burstiness, and offers some performance gains.

Recommendation: Unlimited byte counting is not recommended. Van Jacobson cautions against byte counting [TCPSATMIN] because it leads to burstiness, and recommends ACK spacing [ACKSPACING] instead.

ACK spacing requires ACKs to consistently pass through a single ACK-spacing router. This requirement works well for W-WAN environments if the ACK-spacing router is also the base station.

Limited byte counting warrants further investigation before we can recommend this proposal, but it shows promise.

4.3.2.2 ACK-every-segment

The main idea behind ACK-every-segment is:

- Keep a constant stream of ACKs coming back by turning off delayed ACKs [RFC1122] during slow start. ACK-every-segment must be limited to slow start, in order to avoid penalizing asymmetric-bandwidth configurations.

Recommendation: Implement this on devices but continue investigating. Even though simulations confirm its promise (it allows receivers to receive the second segment from unmodified senders without waiting for a delayed ACK timeout of 200-500 milliseconds), for this technique to be practical the receiver must acknowledge every segment only when the sender is in slow start. Continuing to do so when the sender is in congestion avoidance may have adverse effects on the mobile device's battery consumption and on traffic in the network.

This violates a SHOULD in [TCPCONG]: delayed acknowledgements SHOULD be used by a TCP receiver.

"Disabling Delayed ACKs During Slow Start" is technically unimplementable, as the receiver has no way to know when the sender crosses ssthresh (the "slow start threshold") and begins using the congestion avoidance algorithm. If receivers follow recommendations for increased initial windows, disabling delayed ACKs during an increased initial window would open the TCP window more rapidly without doubling ACK traffic in general.

The conservative recommendation is to ACK only the first segment on a new connection with no delay. Even this conservative recommendation saves one delayed ACK timeout at the receiver, which, in typical WWW applications, saves one delayed ACK timeout in each direction.

Again, much of the benefit of this optimization is also available after the first request-response exchange when clients and servers both implement HTTP/1.1. This optimization works with older servers that implement only HTTP/1.0.

4.3.3 Terminating Slow Start

New mechanisms [ADGGHOSSTT98] are being proposed to improve TCP's adaptive properties such that the available bandwidth is better utilized while reducing the possibility of congesting the network. The latter leads to the closing of the congestion window to 1 segment, and the subsequent slow start phase.

Theoretically, an optimum value for slow-start threshold (ssthresh) allows connection bandwidth utilization to ramp up as aggressively as possible without "overshoot" (using so much bandwidth that packets are lost and congestion avoidance procedures are invoked).

Recommendation: Estimating the slow start threshold is not recommended. Although this would be helpful if we knew how to do it, rough consensus on the tcp-impl and tcp-sat mailing lists is that in non-trivial operational networks there is no reliable method to probe during TCP startup and estimate the bandwidth available.

4.4 ACK Spacing

During slow start, the sender responds to the incoming ACK stream by transmitting two new segments for each ACK received. This results in data being sent at twice the speed at which it can be processed by the network. Accordingly, queues will form, and due to insufficient buffering at the bottleneck router, packets may get dropped before the link's capacity is full. Spacing out the ACKs effectively controls the rate at which the sender will transmit into the network, and may result in little or no queueing at the bottleneck router [ACKSPACING].

Recommendation: No recommendation at this time. Continue monitoring research in this area.

4.5 Delayed Duplicate Acknowlegements

As was mentioned above, link-layer retransmissions may decrease the BER enough that congestion accounts for most of packet losses; still, nothing can be done about interruptions due to handoffs, moving beyond wireless coverage, etc. In this scenario, it is imperative to prevent interaction between link-layer retransmission and TCP retransmission as these layers duplicate each other's efforts. In such an environment it may make sense to delay TCP's efforts so as to give the link-layer a chance to recover [MV97]. It is preferrable to allow a local mechanism to resolve a local problem, instead of invoking TCP's end-to-end mechanism and incurring the associated costs, both in terms of wasted bandwidth and in terms of its effect on TCP's window.

Recommendation: Delaying duplicate acknowledgements may be useful in specific network topologies, but a general recommendation requires further research and experience. Currently, it is not well understood how long the receiver should delay the duplicate acknowledgments.

4.6 Selective Acknowledgements [RFC2018]

SACK may not be useful in many LTNs, according to Section 1.1 of [TCPHP]. In particular, SACK is more useful in the LFN regime, especially if large windows are being used, because there is a considerable probability of multiple segment losses per window. In the LTN regime, TCP windows are much smaller, and burst errors must be much longer in duration in order to damage multiple segments.

Accordingly, the complexity of SACK may not be justifiable, unless there is a high probability of burst errors and congestion on the wireless link. A desire for compatibility with TCP recommendations for non-LTN environments may dictate LTN support for SACK anyway.

[AGS98] recommends use of SACK with Large TCP Windows in satellite environments, and notes that this implies support for PAWS (Protection Against Wrapped Sequence space) and RTTM (Round Trip Time Measurement) as well.

Berkeley's SNOOP protocol research [SNOOP] indicates that SACK does improve throughput for SNOOP when multiple segments are lost per window [BPSK96]. SACK allows SNOOP to recover from multi-segment losses in one round-trip. In this case, the mobile device needs to implement some form of selective acknowledgements. If SACK is not used, recovery from multi-segment losses takes so long that TCP enters congestion avoidance anyway.

Recommendation: Implement SACK now for compatibility with other TCPs and improved performance with SNOOP.

4.7 Detecting Corruption Loss

4.7.1 Without Explicit Notification

In the absence of explicit notification from the network, some researchers have suggested statistical methods for congestion avoidance [Jain89, WC91, VEGAS]. A natural extension of these heuristics would enable a sender to distinguish between losses caused by congestion and other causes. The research results on the reliability of sender-based heuristics is unfavorable [BV97, BV98]. [BV98a] reports better results in constrained environments using packet inter-arrival times measured at the receiver, but highly-variable delay - of the type encountered in wireless environments during intercell handoff - confounds these heuristics.

Recommendation: No recommendation at this time - continue to monitor research results.

4.7.2 With Explicit Notifications

With explicit notification from the network it is possible to determine when a loss is due to congestion. Several proposals along these lines include:

- Explicit Loss Notification (ELN) [BPSK96]

- Explicit Bad State Notification (EBSN) [BBKVP96]

- Explicit Loss Notification to the Receiver (ELNR), and Explicit Delayed Dupack Activation Notification (EDDAN) (notifications to mobile receiver) [MV97]

- Explicit Congestion Notification (ECN) [ECN]

Of these proposals, Explicit Congestion Notification (ECN) seems closest to deployment on the Internet, and will provide some benefit for TCP connections on long thin networks (as well as for all other TCP connections).

Recommendation: No recommendation at this time. Schemes like ELNR and EDDAN [MV97], in which the only systems that need to be modified are the base station and the mobile device are slated for adoption pending further research. However, the requirement that the base station be able to examine the TCP headers flying through it raises the previously stated issues with respect to IPSEC-encrypted packets.

ECN uses the TOS byte in the IP header to carry congestion information (ECN-capable and Congestion-encountered). This byte is not encrypted in IPSEC, so ECN can be used on TCP connections that are encrypted using IPSEC.

Recommendation: Implement ECN when (and if) the IETF finalizes the specification.

Note: Absence of packets marked with ECN should not be interpreted by ECN-capable TCP connections as a green light for aggressive retransmissions. On the contrary, during periods of extreme network congestion routers may drop packets marked with explicit notification because their buffers are exhausted - exactly the wrong time for a host to begin retransmitting aggressively.

4.8 Active Queue Management

As has been pointed out above, TCP responds to congestion by closing down the window and invoking slow start. Long-delay networks take a particularly long time to recover from this condition. Accordingly, it is imperative to avoid congestion in LTNs. To remedy this, active queue management techniques have been proposed as enhancements to routers throughout the Internet [RED]. The primary motivation for deployment of these mechanisms is to prevent "congestion collapse" (a severe degradation in service) by controlling the average queue size at the routers. As the average queue length grows, Random Early Detection [RED] increases the possibility of dropping packets.

The benefits are:

- Reduce packet drops in routers. By dropping a few packets before severe congestion sets in, RED avoids dropping bursts of packets.

- Provide lower delays. This follows from the smaller queue sizes, and is particularly important for interactive applications, for which the inherent delays of wireless links already push the user experience to the limits of the non-acceptable.

- Avoid lock-outs. Packets from over-agressive flows can get dropped with the same probability as other packets.

Active Queue Management has two components: (1) routers detect congestion before exhausting their resources, and (2) they provide some form of congestion indication. Dropping packets via RED is only one example of the latter. Another way to indicate congestion is to use ECN [ECN] as discussed above under "Detecting Corruption Loss: With Explicit Notifications."

Recommendation: RED is currently being deployed in the internet, and LTNs should follow suit. ECN deployment should complement RED's when the IETF finalizes the specification.

4.9 Scheduling Algorithms

Active queue management helps control the length of the queues. Additionally, a general solution requires replacing FIFO with other scheduling algorithms that improve:

1. Fairness (by policing how different packet streams utilize the available bandwidth), and

2. Throughput (by improving the transmitter's radio channel utilization).

For example, fairness is necessary for interactive applications (like telnet or web browsing) to coexist with bulk transfer sessions. Proposals here include:

- Fair Queueing (FQ) [Demers90]

- Class-based Queueing (CBQ) [Floyd95]

Even if they are only implemented over the wireless link portion of the communication path, these proposals are attractive in wireless LTN environments, because new connections for interactive applications can have difficulty starting when a bulk TCP transfer has already stabilized using all available bandwidth.

In our base architecture described above, the mobile device typically communicates directly with only one wireless peer at a given time: the base station. In some W-WANs, it is possible to directly address other mobiles within the same cell. Direct communication with each such wireless peer may traverse a spatially distinct path, each of which may exhibit statistically independent radio link characteristics. Channel State Dependent Packet Scheduling (CSDP) [BBKT96] tracks the state of the various radio links (as defined by the target devices), and gives preferential treatment to packets destined for radio links in a "good" state. This avoids attempting to transmit to (and expect acknowledgements from) a peer on a "bad" radio link, thus improving throughput.

A further refinement of this idea suggests that both fairness and throughput can be improved by combining a wireless-enhanced CBQ with CSDP [FSS98].

Recommendation: No recommendation at this time, pending further study.

4.10 Split TCP and Performance-Enhancing Proxies (PEPs)

Given the dramatic differences between the wired and the wireless links, a very common approach is to provide some impedance matching where the two different technologies meet: at the base station.

The idea is to replace an end-to-end TCP connection with two clearly distinct connections: one across the wireless link, the other across its wireline counterpart. Each of the two resulting TCP sessions operates under very different networking characteristics, and may adopt the policies best suited to its particular medium. For example, in a specific LTN topology it may be desirable to modify TCP Fast Retransmission to resend after the first duplicate ack and Fast Recovery not to shrink TCP cnwd if the LTN link has an extremely long RTT, is known to not reorder packets, and is not subject to congestion. Moreover, on a long-delay link or on a link with a relatively high bandwidth-delay product it may be desirable to "slow-start" with a relatively large initial window, even larger than four segments. While these kinds of TCP modifications can be negotiated to be employed over the LTN link, they would not be deployed end-to-end over the global Internet. In LTN topologies where the underlying link characteristics are known, a great number of similar type of performance enhancements can be employed without endangering operations over the global Internet.

Split-TCP proposals include schemes like I-TCP [ITCP] which achieves performance improvements by abandoning end-to-end semantics. [MTCP] alleviates this problem somewhat, but does not entirely solve it. The Mowgli architecture [MOWGLI] proposes a split approach with support for various enhancements at all the protocol layers, not only at the transport layer. Mowgli provides an option to replace the TCP/IP core protocols on the LTN link with a custom protocol that is tuned for LTN links [KRLKA97]. Also with this option, Mowgli preserves the socket semantics on the mobile device so that legacy applications can be run unmodified.

With LTN links, significant improvements can be achieved at the application layer by introducing application level proxies. The Mowgli system provides full support for adding application level agent-proxy pairs between the client and the server, the agent on the mobile device and the proxy on the LTN-edge device. Such a pair may be either explicit or fully transparent to the applications. Good examples of enhancements achieved with application level proxies include Mowgli WWW [LAKLR95], [LHKR96] and WebExpress [HL96],[CTCSM97].

Berkeley's SNOOP protocol [SNOOP] is a hybrid scheme mixing link-layer reliability mechanisms with the split connection approach. It is an improvement over I-TCP in that end-to-end semantics are retained as well as in terms of performance. SNOOP does two things:

1. Locally (on the wireless link) retransmit lost packets, instead of allowing TCP to do so end-to-end.

2. Suppress the duplicate acks on their way from the receiver back to the sender, thus avoiding fast retransmit and congestion avoidance at the latter.

WTCP [WTCP] is similar to SNOOP in that it preserves end-to-end semantics. In WTCP, the base station uses a complex scheme to hide the time it spends moving packets across the wireless link (this typically includes retransmissions due to error recovery, but may also include time spent dealing with congestion). In order to work effectively, it assumes that the TCP endpoints implement the Timestamps option in RFC 1323 [TCPHP]. Unfortunately, support for RFC 1323 in TCP implementations is not yet widespread. Beyond this, WTCP requires changes only at the base station.

All of these schemes require the base station to examine and operate on the traffic between the portable wireless device and the TCP server on the wired Internet. None of these work if the IP traffic is encrypted, unless, of course, the base station shares the security association between the mobile device and its end-to-end peer. They also require that both the data and the corresponding ACKs traverse the same base station. Furthermore, if the base station retransmits packets at the transport layer across the wireless link, this may duplicate efforts by the link-layer. SNOOP has been described by its designers as a TCP-aware link-layer. This is the right approach: the link and network layers can be much more aware of each other than traditional OSI layering suggests.

Encryption of IP packets via IPSEC's ESP header (in either transport or tunnel mode) renders the TCP header and payload unintelligible to the base station. This precludes SNOOP from working, because it needs to examine the TCP headers in both directions. Possible solutions involve:

- making the SNOOPing base station a party to the security association between the client and the server

- IPSEC tunneling mode, terminated at the SNOOPing base station

However, these techniques require that users trust base stations. Users valuing both privacy and performance should use SSL or SOCKS for end-to-end security.

Recommendation: Implement SNOOP on base stations now. Research results are encouraging, and it is an "invisible" optimization in that neither the client nor the server needs to change, only the base station (for basic SNOOP without SACK).

In some proposals, in addition to a PEP mechanism at the base station, custom protocols are used on the wireless link (for example, [WAP], [YB94] or [MOWGLI]).

Even if the gains from using non-TCP protocols are moderate or better, the wealth of research on optimizing TCP for wireless, and compatibility with the Internet are compelling reasons to adopt TCP on the wireless link (enhanced as suggested in section 5 below).

4.11 Header Compression Alternatives

Because Long Thin Networks are bandwidth-constrained, compressing every byte out of over-the-air segments is worth while.

Mechanisms for TCP and IP header compression defined in [RFC1144, IPHC, IPHC-PPP] provide the following benefits:

- Improve interactive response time

- Allow using small packets for bulk data with good line efficiency

- Allow using small packets for delay sensitive low data-rate traffic

- Decrease header overhead (for the smallest MTU of 512 the header overhead of TCP over IP can decrease from 11.7 to less than 1 per cent.

- Reduce packet loss rate over lossy links.

Van Jacobson (VJ) header compression [RFC1144] describes a Proposed Standard for TCP Header compression that is widely deployed. It uses TCP timeouts to detect a loss of synchronization between the compressor and decompressor. [IPHC] includes an explicit request for retransmission of an uncompressed packet to allow resynchronization without waiting for a TCP timeout (and executing congestion avoidance procedures).

Recommendation: Implement VJ header compression on devices now. It is a widely deployed Proposed Standard. However, it should only be enabled when operating over reliable LTNs, because even a single bit error would most probably result in a full TCP window being dropped, followed by a costly recovery via slow-start.

Implement [IPHC] when the Internet-Draft becomes stable.

4.12 IP Payload Compression

Compression of IP payloads is also desirable. "IP Payload Compression Protocol (IPComp)" [IPPCP] defines a framework where common compression algorithms can be applied to arbitrary IP segment payloads. IP payload compression is something of a niche optimization. It is necessary because IP-level security converts IP payloads to random bitstreams, defeating commonly-deployed link-layer compression mechanisms which are faced with payloads that have no redundant "information" that can be more compactly represented.

However, many IP payloads are already compressed (images, audio, video, "zipped" files being FTPed), or are already encrypted above the IP layer (SSL/TLS, etc.). These payloads will not "compress" further, limiting the benefit of this optimization.

HTTP-NG is considering supporting compression of resources at the HTTP level, which would provide equivalent benefits for common compressible MIME types like text/html. This will reduce the need for IPComp. If IPComp is deployed more rapidly than HTTP-NG, IPComp compression of HTML and MIME headers would be beneficial.

In general, application-level compression can often outperform IPComp, because of the opportunity to use compression dictionaries based on knowledge of the specific data being compressed.

Recommendation: Track IPComp and HTTP-NG standardization and deployment for now.

4.13 TCP Control Block Interdependence [Touch97]

TCP maintains per-connection information such as connection state, current round-trip time, congestion control or maximum segment size. Sharing information between two consecutive connections or when creating a new connection while the first is still active to the same host may improve performance of the latter connection. The principle could easily be extended to LAN coverage rather than limiting itself to hosts. [Touch97] describes cache update for both cases.

Users of W-WAN devices frequently request connections to the same servers or set of servers. For example, in order to read their email or to initiate connections to other servers, the devices may be configured to always use the same email server or WWW proxy. The main advantage of this proposal is that it relieves the application of the burden of optimizing the transport layer. In order to improve the performance of TCP connections, this mechanism only requires changes at the wireless device.

In general, this scheme should improve the dynamism of connection setup without increasing the cost of the implementation.

Recommendation: Recommended for implementation. Monitor research on this.

5 Summary of Recommended Optimizations

The table below summarizes our recommendations with regards to the main proposals mentioned above.

The first column, "Stability of the Proposal," refers to the maturity of the mechanism in question. Some proposals are being pursued within the IETF in a somewhat open fashion. An IETF proposal is either an Internet Drafts (I-D) or a Request for Comments (RFC). The former is a preliminary version. There are several types of RFCs. A Draft Standards (DS) is standards track, and carries more weight than a Proposed Standard (PS), which may still undergo revisions. Informational or Experimental RFCs do not specify a standard. Other proposals are isolated efforts with little or no public review, and little chance of garnering industry backing.

"Implemented at" indicates which participant in a TCP session must be modified to implement the proposal. Legacy servers typically cannot be modified, so this column indicates whether implementation happens at either or both of the two nodes under some control: mobile device and base station. The symbols used are: WS (wireless sender, that is, the mobile device's TCP send operation must be modified), WR (wireless receiver, that is, the mobile device's TCP receive operation must be modified), WD (wireless device, that is, modifications at the mobile device are not specific to either TCP send or receive), BS (base station) and NI (network infrastructure).

The "Recommendation" column captures our suggestions. Some mechanisms are endorsed for immediate adoption, others need more evidence and research, and others are not recommended.

Name	Stability of the Proposal	Implemented at	Recommendation
Increased Initial Congestion Window	RFC 2414 (EXP)	WS	Yes (initial_window=3)
Disable delayed ACKs during slow start	NA	WR	When stable
Byte counting instead of ACK counting	NA	WS	No
TCP Header compression for PPP	RFC 1144 (PS)	WD BS	Yes (see 4.11)
IP Payload Compression (IPComp)	IETF I-D (Approved as PS)	WD (simultaneously needed on Server)	When stable
IP Header Compression (includes IP-IP) for PPP	IETF I-D	WD BS	When stable (For Mobile IP only)
SNOOP plus SACK	In limited use	BS WD (for SACK)	Yes
Fast retransmit/fast recovery	RFC 2001 (PS)	WD	Yes (should be there already)
Transaction/TCP	RFC 1644 (Experimental)	WD (simultaneously needed on Server)	No
Estimating Slow Start Threshold (ssthresh)	NA	WS	No
Delayed Duplicate Acknowledgements	Not stable	WR BS (for notifications)	When stable
Class-based Queuing on End Systems	NA	WD	When stable

Explicit Congestion Notification	IETF I-D	WD NI	When stable
TCP Control Block Interdependence	RFC 2140 (Informational)	WD	Yes (Track research)

Of all the optimizations in the table above, only SNOOP plus SACK and Delayed duplicate acknowledgements are currently being proposed only for wireless networks. The others are being considered even for non-wireless applications. Their more general applicability attracts more attention and analysis from the research community.

Of the above mechanisms, only Header Compression (for IP and TCP) and "SNOOP plus SACK" cease to work in the presence of IPSec.

6 Conclusion

In view of the unpredictable and problematic nature of long thin networks, arriving at an optimized transport is a daunting task. We have reviewed the existing proposals along with future research items. Based on this overview, we also recommend mechanisms for implementation in long thin networks (LTNs).

7 Acknowledgements

The authors are deeply indebted to the IETF tcpsat and tcpimpl working groups, and to Prof. Nitin Vaidya (Texas A&M) whose many insightful comments have proved invaluable in our attempt at improving this note. The following individuals have also provided valuable feedback: Mark Allman (NASA), Raphi Rom (Technion/Sun), Charlie Perkins (Sun), Peter Stark (Ericsson).

8 References

[ACKSPACING] Partridge, C., "ACK Spacing for High Delay-Bandwidth Paths with Insufficient Buffering," September 1998. Internet-Draft draft-rfced-info-partridge-01.txt (work in progress).

[ADGGHOSSTT98] Mark Allman, Spencer Dawkins, Dan Glover, Jim Griner, John Heidemann, Shawn Osterman, Keith Scott, Jeffrey Semke, Joe Touch, Diepchi Tran. Ongoing TCP Research Related to Satellites, August 1998. Internet-Draft draft-ietf-tcpsat-res-issues-04.txt (work in progress).

[AGS98] Mark Allman, Dan Glover, Luis Sanchez. Enhancing TCP Over Satellite Channels using Standard Mechanisms, September 1998. Internet-Draft draft-ietf-tcpsat-stand-mech-06.txt (work in progress).

[Allman98] Allman, M., "On the Generation and Use of TCP Acknowledgments," July, 1998. Submitted to ACM Computer Communication Review.

[AHO98] Allman, M., Hayes, C., Ostermann, S., "An Evaluation of TCP with Larger Initial Windows," Computer Communication Review, 28(3), July 1998.

[BBKT96] Bhagwat, P., Bhattacharya, P., Krishna, A., Tripathi, S., "Enhancing Throughput over Wireless LANs Using Channel State Dependent Packet Scheduling," in Proc. IEEE INFOCOM'96, pp. 1133-40, March 1996.

[BBKVP96] Bakshi, B., P., Krishna, N., Vaidya, N., Pradhan, D.K., "Improving Performance of TCP over Wireless Networks," Technical Report 96-014, Texas A&M University, 1996.

[BPSK96] Balakrishnan, H., Padmanabhan, V., Seshan, S., Katz, R., "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links," in ACM SIGCOMM, Stanford, California, August 1996.

[BV97] Biaz, S., Vaidya, N., "Using End-to-end Statistics to Distinguish Congestion and Corruption Lossses: A Negative Result," Texas A&M University, Technical Report 97-009, August 18, 1997.

[BV98] Biaz, S., Vaidya, N., "Sender-Based heuristics for Distinguishing Congestion Losses from Wireless Transmission Losses," Texas A&M University, Technical Report 98-013, June 1998.

[BV98a] Biaz, S., Vaidya, N., "Discriminating Congestion Losses from Wireless Losses using Inter-Arrival Times at the Receiver," Texas A&M University, Technical Report 98-014, June 1998.

[BW97] Brasche, G., Walke, B., "Concepts, Services, and Protocols of the New GSM Phase 2+ general Packet Radio Service," IEEE Communications Magazine, Vol. 35, No. 8, August 1997.

[CB96] Cheshire, S., Baker, M., "Experiences with a Wireless Network in MosquitoNet," IEEE Micro, February 1996. Available online as: http://rescomp.stanford.edu/~cheshire/papers/wireless.ps.

[CDMA] Electronic Industry Alliance(EIA)/Telecommunications Industry Association (TIA), IS-95: Mobile Station-Base Station Compatibility Standard for Dual-Mode Wideband Spread Spectrum Cellular System, 1993.

[CDPD] Wireless Data Forum, CDPD System Specification, Release 1.1, 1995.

[CTCSM97] Chang, H., Tait, C., Cohen, N., Shapiro, M., Mastrianni, S., Floyd, R., Housel, B., Lindquist, D., "Web Browsing in a Wireless Environment: Disconnected and Asynchronous Operation in ARTour Web Express," in Proc. MobiCom'97, Budapest, Hungary, September 1997.

[Demers90] Demers, A., Keshav, S., and Shenker, S., Analysis and Simulation of a Fair Queueing Algorithm, Internetworking: Research and Experience, Vol. 1, 1990, pp. 3-26.

[ECN] Ramakrishnan, K.K., Floyd, S., "A Proposal to add Explicit Congestion Notification (ECN) to IP", October 1998. Internet-Draft draft-kksjf-ecn-03.txt (work in progress).

[Floyd95] Floyd, S., and Jacobson, V., Link-sharing and Resource Management Models for Packet Networks. IEEE/ACM Transactions on Networking, Vol. 3 No. 4, pp. 365-386, August 1995.

[FSS98] Fragouli, C., Sivaraman, V., Srivastava, M., "Controlled Multimedia Wireless Link Sharing via Enhanced Class-Based Queueing with Channel-State-Dependent Packet Scheduling," Proc. IEEE INFOCOM'98, April 1998.

[GPRS] ETSI, "General Packet Radio Service (GPRS): Service Description, Stage 2," GSM03.60, v.6.1.1 August 1998.

[GSM] Rahnema, M., "Overview of the GSM system and protocol architecture," IEEE Communications Magazine, vol. 31, pp 92-100, April 1993.

[HL96] Hausel, B., Lindquist, D., "WebExpress: A System for Optimizing Web Browsing in a Wireless Environment," in Proc. MobiCom'96, Rye, New York, USA, November 1996.

[IPPCP] A. Shacham, R. Monsour, R. Pereira, M. Thomas, IP Payload Compression Protocol (IPComp), May 1998. Internet-Draft draft-ietf-ippcp-protocol-06.txt (work in progress).

[IPHC] Mikael Degermark, Bjorn Nordgren,Stephen Pink. IP Header Compression, June 1998. Internet-Draft

draft-degermark-ipv6-hc-06.txt (work in progress).

[IPHC-PPP] Mathias Engan, S. Casner, C. Bormann. IP Header Compression over PPP, December 1997. Internet-Draft draft-engan-ip-compress-02.txt (work in progress).

[ITCP] Bakre, A., Badrinath, B.R., "Handoff and Systems Support for Indirect TCP/IP. In Proceedings of the Second USENIX Symposium on Mobile and Location-Independent Computing, Ann Arbor, Michigan, April 10-11, 1995.

[Jain89] Jain, R., "A Delay-Based Approach for Congestion Avoidance in Interconnected Heterogeneous Computer Networks," Digital Equipment Corporation, Technical Report DEC-TR-566, April 1989.

[KRLKA97] Kojo, M., Raatikainen, K., Liljeberg, M., Kiiskinen, J., Alanko, T., "An Efficient Transport Service for Slow Wireless Telephone Links," in IEEE Journal on Selected Areas of Communication, volume 15, number 7, September 1997.

[LAKLR95] Liljeberg, M., Alanko, T., Kojo, M., Laamanen, H., Raatikainen, K., "Optimizing World-Wide Web for Weakly-Connected Mobile Workstations: An Indirect Approach," in Proc. 2nd Int. Workshop on Services in Distributed and Networked Environments, Whistler, Canada, pp. 132-139, June 1995.

[LHKR96] Liljeberg, M., Helin, H., Kojo, M., Raatikainen, K., "Mowgli WWW Software: Improved Usability of WWW in Mobile WAN Environments," in Proc. IEEE Global Internet 1996 Conference, London, UK, November 1996.

[LS98] Lettieri, P., Srivastava, M., "Adaptive Frame Length Control for Improving Wireless Link Throughput, Range, and Energy Efficiency," Proc. IEEE INFOCOM'98, April 1998.

[MNCP] Piscitello, D., Phifer, L., Wang, Y., Hovey, R., Mobile Network Computing Protocol (MNCP), August 1997. Internet-Draft draft-piscitello-mncp-00.txt (work in progress).

[MOWGLI] Kojo, M., Raatikainen, K., Alanko, T., "Connecting Mobile Workstations to the Internet over a Digital Cellular Telephone Network," in Proc. Workshop on Mobile and Wireless Information Systems (MOBIDATA), Rutgers University, NJ, November 1994. Available at: http://www.cs.Helsinki.FI/research/mowgli/. Revised version published in Mobile Computing, pp. 253-270, Kluwer, 1996.

[MSMO97] Mathis, M., Semke, J., Mahdavi, J., Ott, T., "The

Macroscopic Behavior of the TCP Congestion Avoidance Algorithm," in Computer Communications Review, a publication of ACM SIGCOMM, volume 27, number 3, July 1997.

[MTCP] Brown, K. Singh, S., "A Network Architecture for Mobile Computing," Proc. IEEE INFOCOM'96, pp. 1388-1396, March 1996. Available at ftp://ftp.ece.orst.edu/pub/singh/papers/transport.ps.gz.

[MV97] Mehta, M., Vaidya, N., "Delayed Duplicate-Acknowledgements: A Proposal to Improve Performance of TCP on Wireless Links," Texas A&M University, December 24, 1997. Available at http://www.cs.tamu.edu/faculty/vaidya/mobile.html

[NETBLT] John C. White, NETBLT (Network Block Transfer Protocol), draft-white-protocol-stack-00.txt, April 1997 - work in progress.

[RED] Braden, B. Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K.K., Shenker, S., Wroclawski, J., Zhang, L., "Recommendations on Queue Management and Congestion Avoidance in the Internet," RFC 2309, April 1998.

[RFC908] Velten, D., Hinden, R., Sax, J., "Reliable Data Protocol", RFC 908, July 1984.

[RFC1030] Lambert, M., "On Testing the NETBLT Protocol over Divers Networks", RFC 1030, November 1987.

[RFC1122] Braden, R., Requirements for Internet Hosts -- Communication Layers, October 1989.

[RFC1144] Jacobson, V., Compressing TCP/IP Headers for Low-Speed Serial Links, RFC 1144, February 1990.

[RFC1151] Partridge, C., Hinden, R., Version 2 of the Reliable Data Protocol (RDP), RFC 1151, April 1990.

[RFC1191] Jeff Mogul and Steve Deering. Path MTU Discovery, November 1990. RFC 1191.

[RFC1397] Braden, R., Extending TCP for Transactions -- Concepts, November 1992.

[RFC1644] Braden, R., T/TCP -- TCP Extensions for Transactions Functional Specification, July 1994.

[RFC1661] Simpson, W., ed., "The Point-To-Point Protocol (PPP)", RFC 1661, July 1994.

[RFC1986] Polites, W., Wollman, W., Woo, D., Langan, R., "Experiments with a Simple File Transfer Protocol for Radio Links using Enhanced Trivial File Transfer Protocol (ETFTP)", RFC 1986, August 1996.

[RFC2001] W. Richard Stevens. TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms, January 1997. RFC 2001.

[RFC2002] Perkins, C., Editor, "IP Mobility Support", RFC 2002, October 1996.

[RFC2018] Mathis, M., Mahdavi, J., Floyd, S., and Romanow, A., "TCP Selective Acknowledgment Options", October, 1996.

[RFC2188] Banan, M., Taylor, M., Cheng, J., "AT&T/Neda's Efficient Short Remote Operations (ESRO) Protocol Specification Version 1.2", RFC 2188, September 1997.

[RFC2414] Mark Allman, Sally Floyd, Craig Partridge. Increasing TCP's Initial Window, September 1998. RFC 2414.

[RFC2415] Poduri, K., Nichols, K. Simulation Studies of Increased Initial TCP Window Size, September 1998. RFC 2415.

[RFC2416] Tim Shepard and Craig Partridge. When TCP Starts Up With Four Packets Into Only Three Buffers, September 1998. RFC 2416.

[SNOOP] Balakrishnan, H., Seshan, S., Amir, E., Katz, R., "Improving TCP/IP Performance over Wireless Networks," Proc. 1st ACM Conf. on Mobile Computing and Networking (Mobicom), Berkeley, CA, November 1995.

[TCPCONG] W. Richard Stevens, Mark Allman, Vern Paxson. TCP Congestion Control, November 1998. Internet-Draft draft-ietf-tcpimpl-cong-control-01.txt (work in progress).

[TCPHP] Van Jacobson, Robert Braden, and David Borman. TCP Extensions for High Performance, May 1992. RFC 1323.

[TCPSATMIN] TCPSAT Minutes, August, 1997. Available at: http://tcpsat.lerc.nasa.gov/tcpsat/meetings/munich-minutes.txt.

[Touch97] Touch, T., "TCP Control Block Interdependence," RFC 2140, April 1997.

[VEGAS] Brakmo, L., O'Malley, S., "TCP Vegas, New Techniques for Congestion Detection and Avoidance," SIGCOMM'94, London, pp 24-35, October 1994.

[VMTP] Cheriton, D., "VMTP: Versatile Message Transaction Protocol", RFC 1045, February 1988.

[WAP] Wireless Application Protocol Forum. http://www.wapforum.org/

[WC91] Wang, Z., Crowcroft, J., "A New Congestion Control Scheme: Slow Start and Search," ACM Computer Communication Review, vol 21, pp 32-43, January 1991.

[WTCP] Ratnam, K., Matta, I., "WTCP: An Efficient Transmission Control Protocol for Networks with Wireless Links," Technical Report NU-CCS-97-11, Northeastern University, July 1997. Available at: http://www.ece.neu.edu/personal/karu/papers/WTCP-NU.ps.gz

[YB94] Yavatkar, R., Bhagawat, N., "Improving End-to-End Performance of TCP over Mobile Internetworks," Proc. Workshop on Mobile Computing Systems and Applications, IEEE Computer Society Press, Los Alamitos, California, 1994.

Authors' addresses

Questions about this document may be directed at:

Gabriel E. Montenegro
Sun Labs Networking and Security Group
Sun Microsystems, Inc.
901 San Antonio Road
Mailstop UMPK 15-214
Mountain View, California 94303

Voice: +1-650-786-6288
Fax: +1-650-786-6445
E-Mail: gab@sun.com

Spencer Dawkins
Nortel Networks
P.O. Box 833805
Richardson, Texas 75083-3805

Voice: +1-972-684-4827
Fax: +1-972-685-3292
E-Mail: sdawkins@nortel.com

Markku Kojo
University of Helsinki/Department of Computer Science
P.O. Box 26 (Teollisuuskatu 23)
FIN-00014 HELSINKI
Finland

Voice: +358-9-7084-4179
Fax: +358-9-7084-4441
E-Mail: kojo@cs.helsinki.fi

Vincent Magret
Corporate Research Center
Alcatel Network Systems, Inc
1201 Campbell
Mail stop 446-310
Richardson Texas 75081 USA
M/S 446-310

Voice: +1-972-996-2625
Fax: +1-972-996-5902
E-mail: vincent.magret@aud.alcatel.com